Individual Essay by Chensheng Ma:

* **Write a short essay talking about your understanding of transactions, looks and isolation level.**

The transaction usually is a logical work unit that perform one or more activities in a database. Transactions will exist every place and second around us, such as back transaction, message, a blog of twitter, and so on. The transaction may consist of a single CRUD or a combination of them. And every transaction will keep the ACID properties.

Isolation level is used to define the degree to which transaction should be isolated from the resource or data modification made by other concurrent transaction. There are three isolation levels which are Read Uncommitted, Read Committed, Repeatable Read, and Snapshot.

For the Read Uncommitted, there are no lock to prevent other transactions from modifying data read by the current transaction. Also, transactions are not blocked by exclusive locks at the time of data modification, thus allowing other transactions to read the modified data which is not yet committed.

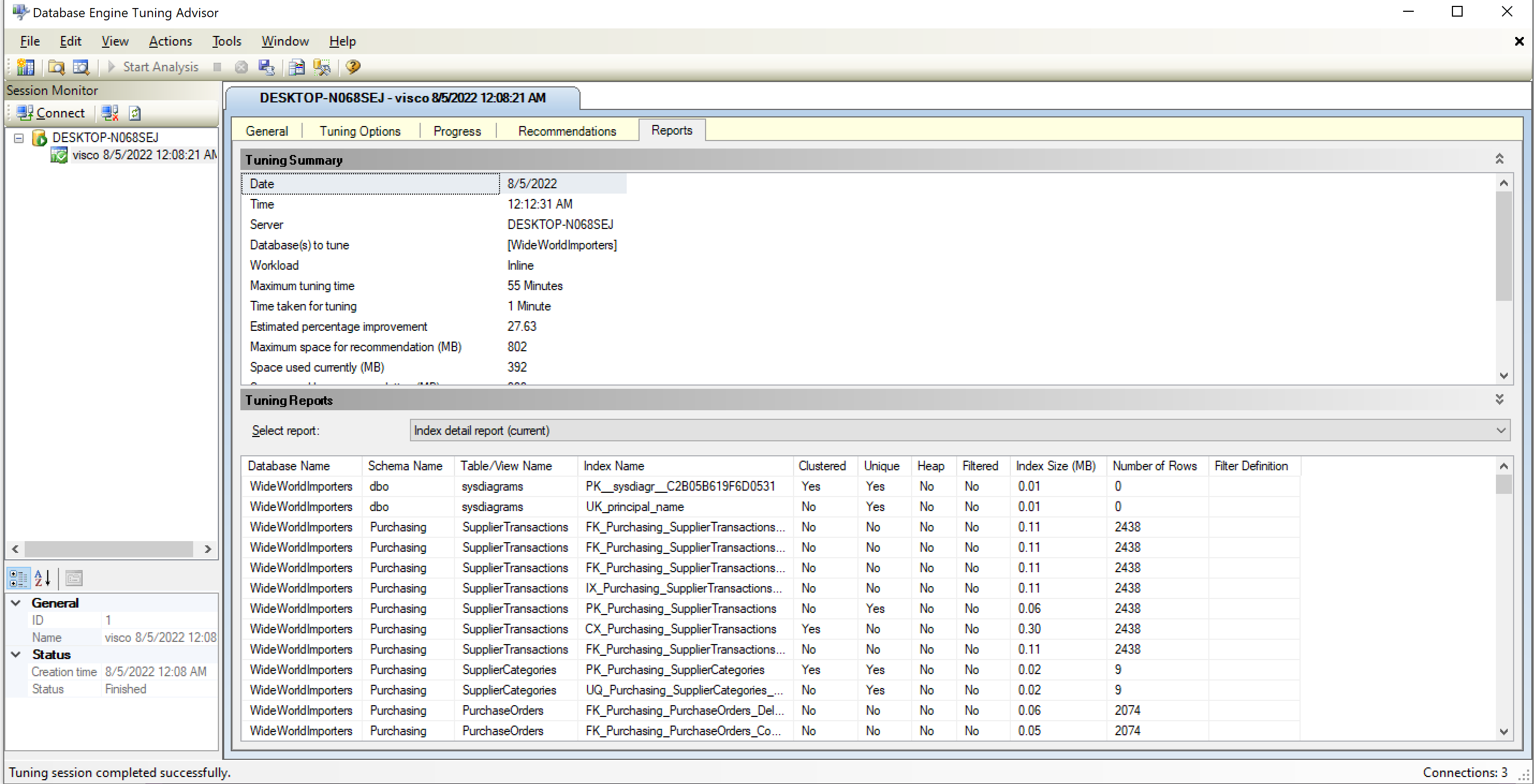
With the Read Committed, transaction issue exclusive lock at the time of data modification.

In Repeatable Read, statements can not read data that has been modified but not yet committed by other transactions. This means no other transaction can modify data that has been read by the current transaction until the current transaction completes.

In the next level of isolation, data read by any statement in a transaction will be the transactional consistent version of the data that existed at the start of the transaction.

The locks can prevent concurrent use of resources by different transactions.

* **Write a short essay, plus screenshot talking about performance tuning in SQL Server. Must include Tuning Advisor, Extended Events, DMV, Logs and Execution Plan.**



Graphical user interface, text, application, email

Description automatically generated

Through the above screenshot, we can notice that the clustered index processes the whole resources. This better than a full table scan. No other operation existed in our operation. I think the current statement is good enough.

* **Write a short essay talking about a scenario: Good news everyone! We (Wide World Importers) just brought out a small company called “Adventure works”! Now that bike shop is our sub-company. The first thing of all works pending would be to merge the user logon information, person information (including emails, phone numbers) and products (of course, add category, colors) to WWI database. Include screenshot, mapping and query.**

In the begining of the merge, we have to insert our bike products' info into the Warehouse.StockItems table. And then, we can assign these bike products into their group in StockGroups table.

INSERT INTO Application.People (

FullName, PreferredName, SearchName, IsPermittedToLogon,

LogonName, PhoneNumber, EmailAddress, FaxNumber, LastEditedBy,

ValidFrom, ValidTo)

SELECT

aau.FullName, aau.PreferredName, aau.SearchName, aau.IsPermittedToLogon,

aau.LogonName, aau.PhoneNumber, aau.EmailAddress, aau.FaxNumber,

aau.LastEditedBy, aau.ValidFrom, aau.ValidTo

FROM [Adventure\_works].Application.Users aau

--Insert basic new product info into WWI database

INSERT INTO Warehouse.StockItems (

StockItemID, ColorID, StockItemName, Size, TaxRate, UnitPrice, Tags)

SELECT

aap.Product\_id, aap.ColorID, aap.product\_name,

aap.size, aap.TaxRate, aap.UnitPrice, aap.Category

FROM [Adventure\_works].Application.Products aap

-- Assign product group into WWI stock\_group. Firstly, a new StockGroupID, Stockname, LastEditedBy, ValidFrom, ValidTo info need in the [WWI].Warehouse.StockGroups needed.

INSERT INTO Warehouse.StockGroups(StockGroupID, StockGroupName, LastEditedBy, ValidFrom, ValidTo)

VALUE (11, 'Bike', 1, '2022-08-03 00:00:00.0000000', '9999-12-31 23:59:59.9999999')

--Then we can assign different bike products into their groups in the StockItemStockGroup table

INSERT INTO Warehouse.StockItemStockGroups

SELECT DISTINCT aap.Product\_id, 11, 1, '2022-08-03 00:00:00.0000000'

FROM [Adventure\_works].Application.Products aap

* **Database Design: OLTP db design request for EMS business: when people call 911 for medical emergency, 911 will dispatch UNITs to the given address. A UNIT means a crew on an apparatus (Fire Engine, Ambulance, Medic Ambulance, Helicopter, EMS supervisor). A crew member would have a medical level (EMR, EMT, A-EMT, Medic). All the treatments provided on scene are free. If the patient needs to be transported, that’s where the bill comes in. A bill consists of Units dispatched (Fire Engine and EMS Supervisor are free), crew members provided care (EMRs and EMTs are free), Transported miles from the scene to the hospital (Helicopters have a much higher rate, as you can image) and tax (Tax rate is 6%). Bill should be sent to the patient insurance company first. If there is a deductible, we send the unpaid bill to the patient only. Don’t forget about patient information, medical nature and bill paying status.**

Basically, we need a fully normalized schema for this OLTP database design as the diagram shown below. A UNIT can have many crew members, a patient has one unit. A patient can have one insurance company, a company can has many patients. A patient can also have many bills, but a bills only have one patient.

